OXFORD SEMICONDUCTOR

Application Note

Incompatibility between Legacy Base Addresses & PCI Express® Devices

Legacy Base Addresses

In legacy systems, particularly those based on the ISA bus, devices such as UARTs and parallel ports used to reside at fixed base addresses. The addresses were typically allocated as shown in Table 1, although variations on port and base address allocation could occur:

Table 1 Legacy UART Base Addresses

Base Address	Port
3F8	COM1 UART
2F8	COM2 UART
3E8	COM3 UART
2E8	COM4 UART
3BC	LPT1 non-ECP/EPP parallel port often incorporated on video cards
378	LPT1/2 parallel port
278	LPT2/3 parallel port

Although most modern operating systems and software are not constrained by the requirement and can now handle ports at any base address, on a PCI-based system it was still possible to remap such ports to reside at the same base addresses, which removed potential legacy compatibility issues. However, if ports are provided using a PCI Express interface on standard PC systems, it is no longer possible to remap base addresses into this legacy range.

This limitation is not unique to Oxford Semiconductor devices, it is inherent in PCI Express® system architectures. See "Technical Details" on page 2 for further information.

Implications

The limitation is not a major issue in any modern operating system. It seems not to cause any significant issue in standard UART or parallel port device drivers. For example, when running in legacy UART mode, the Oxford Semiconductor OXPCIe952 can use the standard Microsoft Windows UART driver. Similarly, the parallel port on the OXPCIe840 or OXPCIe952 can use the standard Microsoft Windows parallel port driver. Operation is fully plug-and-play, and no base-address allocation issues occur.

Known Issue

The only known issue occurs with some specific parallel port devices; not in the driver for the parallel port itself, but in the driver for the device attached to the parallel port, so this is out of the control of Oxford Semiconductor.

Symptom

The symptom of the problem is that although the parallel port, the parallel port device driver and the parallel port device itself fully support ECP or EPP modes of operation, the device only operates in the lower speed SPP mode.

Cause

This problem occurs when the device driver for the device attached to the port tries to determine whether ECP or EPP mode is supported. Some old legacy device drivers are known to check explicitly that a parallel port is at 378 or 278 before enabling ECP or EPP mode, because the old legacy base address of 3BC did not support ECP or EPP.

Any device driver which correctly negotiates the mode of operation, checks the size of memory window for the device to determine supported modes (only four bytes are required if ECP and EPP are not supported, rather than eight), or even checks to see whether the base address of 3BC is used, would work correctly.

Where possible, the issue can easily be avoided by minor changes in the device driver for the device attached to the parallel port. Even if the problem is encountered, the device still works if the lower-speed SPP mode of operation is used.

Technical Details

PCI Express-based devices cannot be mapped onto legacy address ranges because of the way PCI-to-PCI bridges work and how they support legacy devices. This also applies to PCI Express root ports, because they are implemented as a virtual PCI-to-PCI bridge.

PCI-to-PCI bridges (or virtual bridges in PCI Express) can have three types of address decoding to claim I/O cycles and forward them to their secondary bus:

- Subtractive decode—address decoding in which a bridge accepts all access requests that are not positively decoded by another device (after a four-cycle timeout)
- Positive decode—address decoding in which a bridge forwards requests from the primary side to the secondary side of the bridge within a window assigned by the firmware or by the operating system

- Negative decode—address decoding in which a bridge forwards access requests that are outside the assigned positive decode window of the bridge from the secondary side to the primary side. This type of decoding occurs when a device on the secondary side of the bridge attempts to access a device on the primary side of the bridge

The I/O decode window is a subrange of the system I/O space. It is a multiple of 4 Kbytes in size and aligned on natural boundaries (that is, it starts at address X000h and ends at XFFFh).

As a result, it is worth considering how to remap the base address to the legacy range in positive and subtractive decode scenarios. There are several options:

- Positive decode—to overcome the problem for a device such as a parallel port, it might be possible to change the I/O range to a standard range; for example, by using the LogConfigOverride directive in the INF file and writing the range to the device BARs in the driver. For positive decode it is also necessary to change the bridge I/O decode window to 0000-0FFFh, so as to be able to receive I/O cycles on the PCI Express device positively. However, this is not possible because the bridge will then claim all the I/O commands to the lower 4 Kbytes of I/O space, while some commands will also be claimed by another hard-wired motherboard device, such as the timer, IDE, VGA, PIC or DMA controller; and they will both try to assert DevSel#. Such a condition causes a blue screen, so positive decoding cannot be used
- Subtractive decode—when a bridge uses subtractive decode, it forwards any I/O cycles, disregarding the bridge I/O window, to its secondary bus if they are not claimed by another device on the primary bus within four cycles. So if an I/O command is sent to an I/O address such as 0x378 and no in-built parallel port is residing at the address (that is, the command is not claimed by another device), it is claimed by the bridge. The bridge then forwards the cycle to its secondary bus and the PCI parallel device can claim it. This happens when using an OXPCI954 or OXPCI840 on standard addresses, for example

Only one bridge on each bus can use subtractive decode, otherwise two bridges will claim I/O cycles that are not claimed on the primary bus, once again resulting in a blue screen.

There is a fundamental reason why subtractive decode, which works for PCI devices, cannot be used for PCI Express devices. In modern system architectures, such as those using ICH7 (Intel I/O Control Hub 7) or ICH8, the PCI-PCI bridge and PCI Express root ports reside on the same bus (Bus 0). This effectively means that either port can use subtractive decode, but not both. In fact, to support current PCI legacy devices, it is the PCI-to-PCI bridge which uses subtractive decode and PCI Express root ports are hard-coded as positive-only decoders. For this reason, it is impossible to map PCI Express devices such as the OXPCIe840 to use a standard address range.